Artificial Neural Network Approach for NMR Data Analyses

E.R. de Blouw (9102337)
Master Thesis no. 378
Department of Real-time Systems
Mathematics and Informatica Faculty
Nijmegen University
September 8, 1996

Artificial Neural Network Approach for NMR Data Analyses

Master Thesis No. 378

by:

E.R. De Blouw

supervisors:

Dr. Th.E. Schouten
Computing Science Institute
Mathematics and Informatica Faculty
Nijmegen University
&
Dr. Ir. L.J.Th.O. van Erning
Institute of Radiology
University Hospital Nijmegen St. Radboud

September 8, 1996

Preface

As with most things in live, we never actually finish something. This certainly goes for this thesis, which happens to be the end of my student-ship. I started this experiment with the aim to explore the possibilities of neural networks with magnetic resonance data. Along the way I crossed many interesting fields, and they took a great deal of my time.

This project was my first introduction to the world of the Nuclear Magnetic Resonance, which is one of the most interesting parts of this project, not only for the sake of knowing, but also for the obvious simplicity in which we, as being build out of several nuclei, manage to combine certain chemical reactions to an output, that goes far beyond our imagination. The NMR brought me back to the very basics and sometimes raised questions about the essence or non-essence of being, but since this is not a philosophical thesis I wont get into that.

I would like to thank my supervisors Dr. Th.E. Schouten and Dr. Ir. L.J.Th.O. van Erning for the fact that they made this project possible, and of course for their supervision and bright views on lots of issues. Special thanks are directed to the members of the Spectroscopy group of the University Hospital Nijmegen St. Radboud, which always found the time to answer some of my boring questions, and all others for their so called 'moral' support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Based on the patients complaints the treating physician will perform a medical examination. The physician will analyze the results of the medical examination and make a diagnosis. Based on that diagnosis a certain treatment will be chosen. After the treatment it's necessary to check the effect of the treatment on the patients health. If the treatment has little or no effect, the policy has to be adjusted. This cyclic process is part of almost any medical application, especially in areas as oncology where the medical science doesn't have answers for all the questions.

Different diseases are often characterized by changes in metabolism of tissue from various, usually unknown composition. It is often not known to what degree the changes contribute to the illness, and at what locations the metabolite changes take place. It is therefor difficult to evaluate the direct effect of a treatment.

Magnetic Resonance Spectroscopy (MRS) is a unique method to obtain information about metabolic processes.

The complete application of MRS in a clinical examination requires a number of items:

- information about tissue regions with a high as possible resolution
- information about quantitative levels of metabolism
- clinical interpretable results

Quantification of spectra has to be integrated with Spectroscopic Imaging (SI). A combined ^{1}H and ^{31}P 3D SI of the human brain can easily contain 1600 spectra, each containing information of at least 5 metabolites. Clinical evaluation requires the following:

- adequate and reliable quantification of spectrum components in relation with tissue location and voxel composition

- the recognition of normal pathological spectral properties
- a usable clinical representation of the evaluated data

Previous investigation showed that the application of artificial neural networks offers opportunities in the area of:

- peak detection and objective classification of peaks
- classification of spectra belonging to a specific pathological anatomy

MRS is already part of the experimental clinical examinations, and the progress within spectroscopy indicates the major importance of this instrument in future research.

It's however necessary that the results obtained with MRS can be analyzed in an adequate way. The application of artificial neural networks is expected to be one of the possibilities to interpret nuclear magnetic resonance (NMR) data fast, robust and reliable.

The use of neural networks doesn't have to be a straight forward process in which the spectral data are given to the neural network. Figure 1.1 shows the possible steps that have to be taken to come to a final diagnosis. In the ideal case at least the preprocessing part wont be necessary.

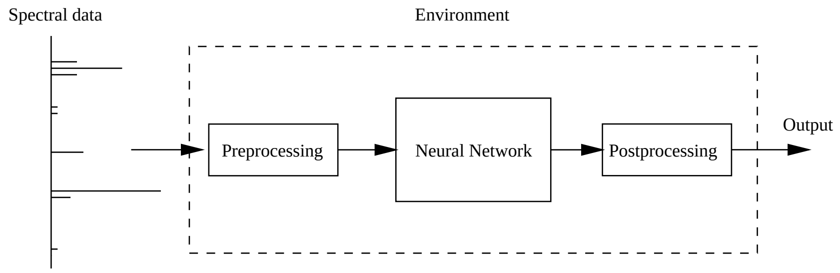


Figure 1.1: The general steps that have to be taken to come to a final output with the use of a neural network.

Part of this thesis deals with the investigation to find out whether preprocessing of spectral data is required or not.

Chapter 2

Introduction to Nuclear Magnetic Resonance

2.1 Introduction

Nuclear magnetic resonance (NMR) is known for a long time, but there was no significant application until 1945.

In 1945 two independent groups, Bloch at Stanford and Purcell at Harvard, observed the magnetic resonance phenomenon, Bloch reported resonance in liquid water while Purcell reported resonance in solid paraffin. Since then, NMR plays an important role in the identification of molecular structures.

The aim of this chapter is not to provide all the basic principles about NMR, but it should provide the reader with enough information to understand what a NMR signal is and what problems arise when we want to use NMR for any special purpose.

For reasons of simplicity only nuclear species with total spin number $I = \frac{1}{2}$ like ^{1}H , ^{13}C , ^{31}P will be considered.

2.2 Nuclear Magnetic Resonance

Nuclei of atoms with an odd number of protons or neutrons possess a non-zero spin quantum number. Following the laws of electromagnetism a moving charge constitutes a current and therefore generates a magnetic field in its neighborhood, causing the individual nuclei to behave as though they possess a *magnetic moment* μ.

2.2.1 Nuclear Precession

If a spinning nucleus is exposed to a static magnetic field B_0 it behaves like a gyroscope in a gravitational field, i.e. it starts precessing about B_0 , with a frequency v_0 which is known as the *Larmor frequency* of the nucleus.(see figure 2.1)

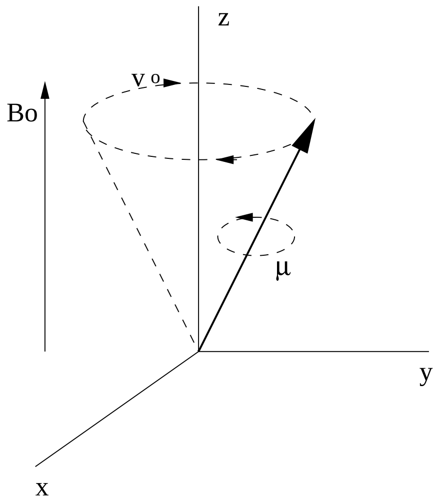


Figure 2.1: Larmor precession of a spinning nucleus in a static magnetic field

One elementary property of spinning nuclei is that they never align completely parallel to the direction of the static magnetic field B_0 , no matter how strong this field is. Instead, the spinning nucleus precesses only faster by increasing the strength of B_0 . The relation between the Larmor frequency and the static magnetic field can be expressed using the equation

$$v_0 = \gamma B_0$$

where v_0 is expressed in radians per second, γ is the gyro magnetic constant of the nucleus and B_0 represents the static magnetic field.

2.2.2 The Nuclear Magnetic Resonance Phenomenon

Following quantum mechanical rules, a nucleus with total spin quantum number I may occupy $(2I + 1)$ different energy levels in a magnetic field. Let m represent the energy level, then for nuclei with $I = \frac{1}{2}$, e.g. ^{1}H , ^{13}C , ^{31}P , two energy levels are possible, m = -$\frac{1}{2}$ or $m = +\frac{1}{2}$.(see figure 2.2)

Precessing of the nuclear spins about B_0 is energetically favored since the component of their magnetic moment μ in the direction of B_0 reinforces the

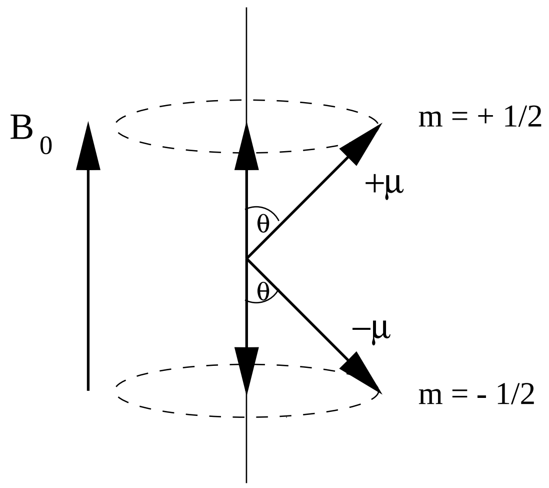


Figure 2.2: Alignment of spinning nuclei in a static magnetic field

magnetic field. (see figure 2.3) Following Schild's[1] explanation, it's easier to walk on your feet then on your hands, the latter needs a different energy state.

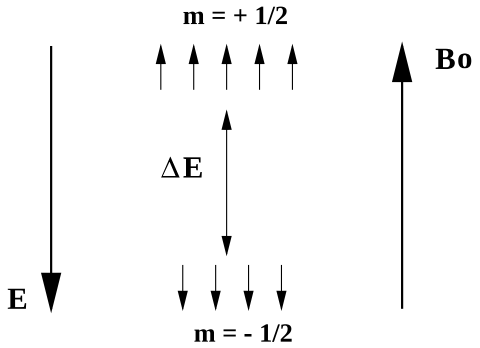


Figure 2.3: Distribution of spin population, the lower energy state is more favorable then the higher energy state

The difference between the energy levels is called ΔE . The nuclear spins can be made to flip over by appending a second magnetic field B_1 , the so called RF field (radio frequency field) , orthogonal to B_0 . If B_1 is made to rotate exactly at the precession frequency v_0 it will cause large oscillations in the angle between μ and B_0 . If the rate of rotation of B_1 is varied through the value of the precessing frequency, which means applying an amount of energy $E \geq \Delta E$, a resonance phenomenon can be observed, e.g. spins are flipping from the lower to the upper energy state.

When the RF field is turned off, the nuclei return to their equilibrium state, meaning that spins return to their original state. The processes responsible for this behavior are relaxation processes.

2.3 Relaxation

The behavior of nuclei in a magnetic field is usually expressed in terms of the magnetic moments, where the net magnetic moment is aligned along the z-axis. Figure 2.3 shows how the magnetization vector M is split into a M_x, M_y and M_z component.

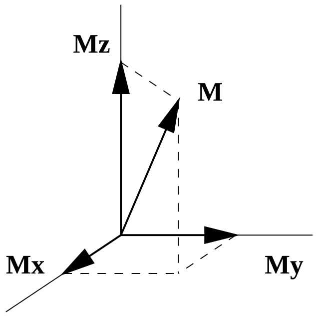


Figure 2.4: The magnetization vector M, split in a M_x, M_y and M_z component.

It's not hard to imagine that when we're dealing with a lot of nuclei spinning in a static magnetic field the net magnetic moment M_{net} will be aligned along the direction of the static magnetic field B_0 , e.g. the z-direction. Due to the spinning of the nuclei the sum of the M_x and M_y components of the magnetic vectors M cancel each other out in the xy-plane, and because the lower energy state is more favorable then the upper energy state, more spins, e.g. magnetic moments, will be in the lower energy state, resulting in a net magnetization in the direction of the static magnetic field B_0 .

The evolution of the nuclear magnetic moment after applying an excitation pulse can be described by the Bloch equations. (see [2] or [5] for a detailed description).

2.3.1 Spin-Lattice Relaxation

At resonance the RF field B_1 causes a spin transfer from the lower to the upper energy state. The equilibrium distribution of the spins in the static magnetic field B_0 is disturbed. Following any disruption the nuclear spins relax to the equilibrium with their surroundings, the 'lattice', which includes B_0 . This relaxation is assumed to be a first order rate process with a constant T_1 characterizing the nucleus. T_1 is called the *spin-lattice relaxation* or *longitudinal relaxation* time, since it is related to the realignment of the magnetic vector along the magnetic field direction. In most cases T_1 can be seen as the lifetime of a nucleus in a certain spin state.

T_1 depends on tissue composition, structure and surrounding. The relaxation process is a process of exchanging thermal energy from the spins to the lattice. The exchange of energy is difficult when the lattice of the nucleus contains small and fast moving molecules. For example, the exchange of energy of protons in pure water takes a relative long time, because the water molecules are small and move too fast, resulting in a long T_1 . The T_1 of fat, containing larger molecules, on the other hand is very short.

Typical values for T_1 in human tissue are in the range from 300 to 2000 ms.

In a spin-lattice relaxation experiment excitation is preceded by a preparation pulse or pulse train and the NMR signal is recorded as a function of the interval between the preparation and the observation pulses.

If the preparation and excitation/observation pulses are not set precisely at 90^o or 180^o and/or if the resonance condition is not matched exactly, modifications have to be made. In practice the static B_0 and RF fields B_1 will both be inhomogeneous over any finite sample volume, and necessitate modification of the simple treatment.

2.3.2 Spin-Spin Relaxation

In solids and liquids with slowly tumbling molecules inter nuclear dipole-dipole interactions may become important. Furthermore, energy quanta are exchanged between nuclei to a certain degree. Both factors tend to shorten the life-times of all spin-states. This second kind of first-order relaxation process is called *spin-spin relaxation* or *transverse relaxation* and is characterized by the time constant T_2 . Due to inhomogeneous magnetic fields and interactions with other nuclei T_2^* is effective instead of T_2 . If we're dealing with a more complex pulse sequence, for example a spin-echo sequence, T_2^* refers to the exponential decay of the transverse magnetization. By generating echo signals from 180^o pulses the real T_2 can be determined.

The relation between T_2 and T_2^* can be expressed using the following formula:

$$\frac{1}{T_2^*} = \frac{1}{T_2} + \gamma \Delta B$$

Where γ is the gyro magnetic ratio of the nucleus observed and ΔB represents both field inhomogeneities and interactions between other nuclei. Figure 2.5 shows the RF-signals of a pulse sequence: upperline the transmitted signal, lower line, the received signal, showing the signal decay related to T_2 and T_2^* .

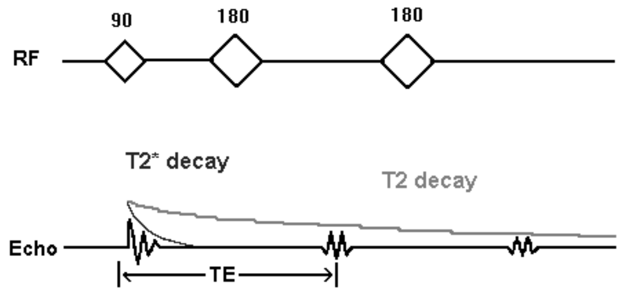


Figure 2.5: The relation between T_2 and T_2^* .

T_2 is related to the time spinning nuclei are in phase and it depends on inhomogeneities of the external magnetic field and inhomogeneities of the local magnetic fields in the tissues.

For example, in liquid water the molecules move very fast, the local magnetic fields fluctuate very fast and sort of average each other out, resulting in small net differences in magnetic field strength inside the tissue, hence the nuclei stay in phase for a relative long time and thus T_2 is long.

Typical values for T_2 are in the range of 30 to 150 ms. T_2 values are usually between two and ten times less then the T_1 values.

2.3.3 Simple Pulse Experiment

Figure 2.6 shows the relaxation process of a simple 90^o pulse experiment. Figure 2.6a shows the longitudinal magnetization along the direction of the static magnetic field. After applying a 90^o RF pulse the longitudinal magnetization decreases and becomes zero, resulting in a transverse magnetization. (figure 2.6b) Applying a 90^o pulse has two major effects:

1. The longitudinal magnetization decreases to zero
2. The spins are precessing in phase, causing the appearance of a magnetization vector called transverse magnetization, precessing in the xy-plane.

After the RF pulse is switched off the longitudinal magnetization increases and the transverse magnetization disappears due to de-phasing of the spins. (figure 2.6c) Both processes are due to entirely different mechanisms, and

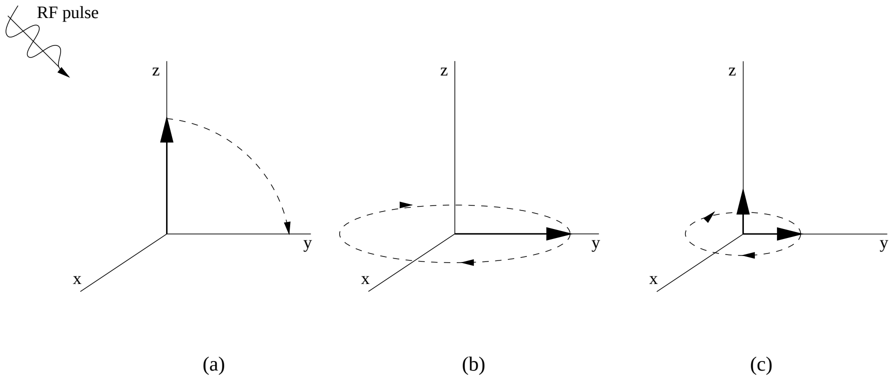


Figure 2.6: A simple 90^o experiment, (a) the net magnetization turns from the z to the xy-plane, (b) longitudinal magnetization is zero and the transverse magnetization is at its maximum value, (c) after the RF pulse is switched off, the longitudinal magnetization increases and returns to its original state, while transverse magnetization decreases to zero (de-phasing).

occur independently even though at the same time. One has to keep in mind that $T_2 \ll T_1$. The sum vector of the magnetization vectors performs a spiraling motion as depicted in figure 2.7

The pulse experiment described here is a very simple one. There are other more complex pulse experiments like for example spin-echo sequences in which one gives a 90^o pulse followed by a 180^o pulse to get the spins back into phase and perform T_2 measurements and of course these sequences can be repeated to obtain as much information as required.

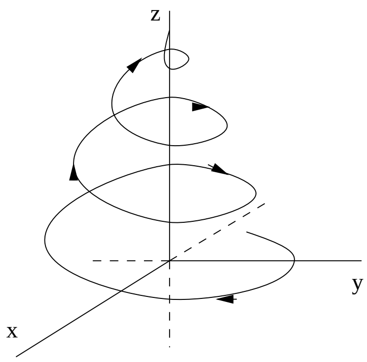


Figure 2.7: The spiraling motion of the total magnetization vector

2.3.4 The Free Induction Decay (FID) signal

Applying a 90^o excitation pulse will cause the net magnetism to move from the z to the xy-plane. The individual nuclear magnetic moments of the nuclei still precess about the z-direction but simultaneously spin-lattice and spin-spin relaxation takes place to restore the net magnetic moment in the z-direction.

The resulting signal, induced in the receiver coil, will be a damped oscillation.(see figure 2.8) If phase detection is used, the received signal will be a difference signal of the received signal and the RF field. At resonance this difference signal should be zero, which provides a method of attaining the resonance condition.

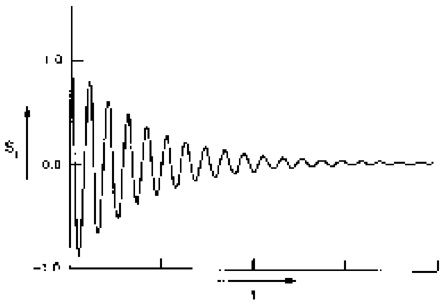


Figure 2.8: Example of a FID of a single signal

However, unless the sample examined is a single molecular species with only a single chemical environment for the resonant nuclei, this beat or difference signal is a superposition of the signals from resonant nuclei in the different chemical sites.

The basic principal sources for the different resonance conditions are:

- A nucleus is shielded by electrons surrounding it, which provides an electronic environment that is characteristic for a site.
- Each nucleus is in the neighborhood of other nuclei which, if they possess magnetic moments, will produce a field at the nucleus under observation and thus will modify the field applied.

2.4 NMR Parameters

The finally recorded signal depends on a number of parameters which will not be explained in detail, but a short description might help to understand the influence of these parameters on the final recorded signal.

T_1 the spin-lattice relaxation, the time at which 63% of the original value of the longitudinal relaxation is reached. This value is specific for tissue components and related to a specific nucleus.

T_2 the spin-spin relaxation, the time when transverse relaxation decreased until 37% of the original value. T_2 differs per tissue components relating to specific nucleus.

TR is the time at which a pulse experiment is repeated

TE is the time to echo, by applying a 180^o pulse. This TE is important when you want to obtain T_2 measurements, since the echo is responsible for restoring some of the intensity of the transverse relaxation signal.

Specially TR and TE determine the final signal intensity and are important parameters for both Magnetic Resonance Imaging (MRI) and Magnetic Resonance Spectroscopy (MRS).

2.5 References

[1] Prof. Dr. Hans H. Schild
MRI made easy (...well almost)
Schering AG Berlin/Bergkamen 1990

[2] William Derbyshire
Introduction to the Physical Basis of NMR
Magnetic Resonance Spectroscopy in Biology and Medicine, p3-p29 (1992)

[3] E. Breitmaier, W. Voelter
^{13}C NMR Spectroscopy
Verlag Chemie 1974

[4] Frank A. Bovey
High Resolution NMR of Macromolecules
Academic Press, New York and London 1972

[5] Richard R. Ernst, Geoffrey Bodenhausen, Alexander Wokaun
Principles of Nuclear Magnetic Resonance in One and Two Dimensions
Oxford university Press 1990

Chapter 3

NMR-Spectroscopy

3.1 Meaning of a NMR-Spectrum

In its most simple form a NMR-Spectrum is a one-dimensional plot of the radio frequencies that the nuclei, once they're properly excited, emit. The reason we need such a plot is that the detailed knowledge of the emission pattern will tell us about the chemical environment as well as the dynamics of the nuclei forming the molecules in the sample.

The frequencies nuclei can absorb are related to the effective magnetic field they experience by the following formula:

$$v = \frac{\gamma}{2\pi}(B_0 - \sigma B_0)$$

where γ is again the gyro magnetic ratio, and σ is the screening factor. The parameter γ undergoes large variations from nucleus to nucleus, the screening factor σ is less sensitive. In practice this means that resonance frequencies for various nuclear species are spread over a wide range of frequencies while for each nuclear species, depending on the chemical environment, there will be a number of resonance frequencies differing slightly from one another. One should think for various nuclear species in the order of several millions of Hz and for single nuclear species in the order of a few thousands Hz.

3.2 The Fourier Transform

The FID is a convolution of radio frequencies decaying exponentially over a period of time. This FID contains all the information concerning the statics and dynamics of the nuclei examined. However, the study of a FID signal is not very straightforward. In order to extract and analyze all the information available special methods have to be used. One of these methods is the Fourier Transform which transforms a time-domain signal into the frequency-domain, which means all components are represented as a function of their frequencies.

In most NMR experiments the FID is recorded using quadrature phase-detection, this means that the signal coming from the net magnetic moment vector is recorded in both x and y direction representing the real- and imaginary part of the FID. The FID can be expressed in the following way:

$$M(t) = M_x(t) + iM_y(t),$$

where $M(t)$ represent the complex magnetization at time t. The complex time signal $s(t)$ is proportional to the complex magnetization, and it is this time signal that can be subjected to a Fourier transformation.

$$S(\omega) = \int_0^\infty s(t)e^{-i\omega t}dt,$$

where $\omega = 2\pi f$.

Since the data are stored in a digitized form, the Fourier transform applied on the complex signal is the discrete Fourier transform (DFT) or its equivalent, the fast Fourier transform (FFT). The DFT is defined by:

$$S[k] = \sum_{n=0}^{N-1} s[n]e^{-2j\frac{\pi kn}{N}},$$

Note that the continuous variable t is replaced by the discreet variable n, where $0 \leq n \leq N-1$.

Figure 3.1 shows an example of a FID with a single frequency and one showing the convolution of two time signal with different frequencies.

3.3 NMR Measurements and sensitivity

The sensitivity of NMR spectroscopy depends on several factors, some of which are intrinsic to nuclear species and others that are extrinsic depending on the sample and/or the instruments.

3.3.1 Intrinsic factors

When we irradiate a sample with a radio frequency pulse there will be both absorption and emission, e.g. transfer from the lower to the upper energy state and vice versa. The intensity of the NMR signal for spins with $I = \frac{1}{2}$ is

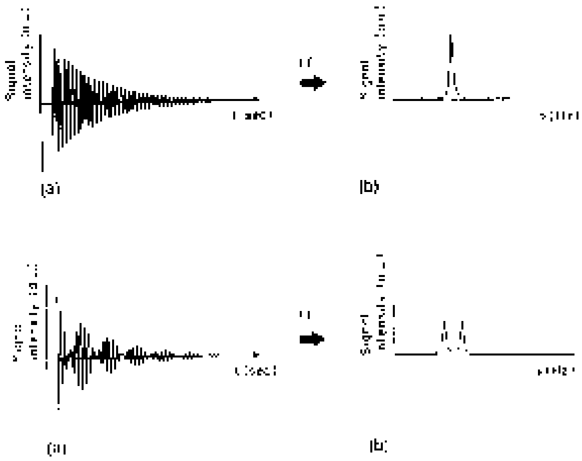


Figure 3.1: The upper figure (a) shows an example of a FID with a single frequency, (b) shows the Fourier transform (FT) of the FID resulting in one peak. The lower figure (a) shows the convolution of two time signals, each with a different frequency, (b) shows again the FT of this signal, this time there are two peaks.

proportional to the ratio of the populations in the two energy states, which is a function of the energy difference ΔE . At thermal equilibrium ΔE will be small and the number of spins in the lower energy state will be slightly greater then in the upper energy state.

A second intrinsic factor is the low isotopic natural abundance of some nuclear species, for example ^{13}C , resulting in a very small percentage of nuclei that will be excitable.

Both factors make it very difficult to detect NMR signals of some nuclei. These problems can be overcome by applying special pulse techniques, and of course by further development of the electronic hardware like dedicated coils, and software, e.g. signal processing techniques.

3.3.2 Extrinsic factors

One of the most important extrinsic factors that determines the sensitivity of a NMR measure is the static magnetic field B_0 . In order to ensure high sensitivity the static magnetic field must:

- be as strong as possible
- have a constant strength in time
- have a high homogeneity

A lack of these factors will cause different resonance frequencies for the nucleus in question, because each nucleus experiences a different magnetic field. This will lead to a broadening of the spectral lines and a decrease in intensity and resolution, which makes it very hard to distinguish nuclei with slightly different magnetic properties.

Other extrinsic factors are the accuracy of the measurement equipment, the amplifiers, coils etc. This equipment contributes to the signal to noise ratio (SNR), which must be obtained as high as possible.

Another factor is the nature of the sample, the presence of for example paramagnetic substances like Fe, Cu, Mn can cause *line broadening*.

The existence of exchange processes taking place at a rate nearby the Larmor frequency can also produce line broadening.

The heterogeneity of the sample also can cause line broadening and therefore reduce sensitivity of the NMR measurement.

3.4 Spectral Improvements

There are a number of reasons why a spectrum taken from a FID does not contain any useful information. Some of these reasons apply to the ones mentioned above, others can be due to the fact that certain parameters were not set correctly.

The following subsection will handle some of the methods that can be used to improve the spectral results, it's beyond the scope of this thesis to get into a detailed description, so they will only be mentioned shortly. (See [1] and [2] for more details)

3.4.1 Phase correction

There are a number of reasons why phase correction is necessary[2]:

- the spectrometer phase sensitive device is not set correctly
- there is a delay between the excitation pulse and the start of the data acquisition, resulting in a truncated FID in the beginning
- there are off resonance effects when the RF amplitude is not large enough compared to the spectral width
- phase errors are introduced when the filter bandwidth is about equal or smaller than the spectral width

The phase can be constant over the spectrum, it can also be linear or have a more complex structure. In the first case a zero-order phase correction can be used, the others require more complex correction methods which will not be described here.

3.4.2 Baseline Correction, Peak Positions and Amplitudes

The obtained spectrum contains background signals that disturb the exact position of the *baseline*. Baseline correction tries to subtract the background signal from the spectrum, using some sort of fitting procedure. These procedures have to be taken with care, since they can affect the peaks of interest in various ways.

Once a spectrum is obtained, the peak positions and amplitudes have to be calculated. This is usually done by calculating the position using the carrier frequency or reference line.

3.4.3 Zero filling

The *spectral resolution* is given by the separation between data points, and is determined by the total sampling time. Zero filling consists of extending the FID with a number of zeros. Taking the Fourier transform of the newly created FID results in a smooth interpolation between the spectral data points and improves the appearance of the spectrum.

In the case of quadrature phase detection, the spectrum can be improved by a factor $\sqrt{2}$ when the dataset is zero filled until $2N$ data points. (See [2])

3.4.4 Apodization techniques

Another technique for improving the appearance of the spectrum is the use of *apodization* techniques. These techniques are based on multiplying the FID with an exponential decaying function that starts with the value 1. The result is a smoothing of the FID from its maximum value to zero. The reason for doing this is that most FIDs do not decay to zero during acquisition time. Taking the FFT of the FID results in an undesirable distortion of the resonance line shape. This effect can be corrected using *time domain window functions*.

There are a lot of these window functions available. Two of them are the *matched filter* and the *Lorentz-Gauss Transformation*. (See [1] and [2] for further explanation)

3.5 The Chemical Shift

As mentioned before nuclei having different chemical environments precess at different Larmor frequencies. The shift of Larmor frequencies due to chemical nonequivalence of nuclei in molecules is called the *chemical shift*.

In order to measure chemical shifts one needs a reference compound which is assigned a frequency of shift zero. The chemical shift of chemically equivalent nuclei of a sample may then be measured as the frequency difference.

To compare various measurements from different machines it is necessary to provide an independent scale. To get chemical shift values that are independent of frequency and magnetic field strength one divides the frequency differences by the radio frequency resulting in a term δ.

$$\delta_s = \frac{v_{0s} - v_{0r}}{v_1} 10^{-6}$$

where δ_s is the resulting chemical shift in ppm (= parts per million), v_{0s} is the Larmor frequency of the sample, v_{0r} is the reference Larmor frequency and v_1 is the radio frequency used.

A common reference compound for example ^{1}H and ^{13}C NMR spectra is tetramethylsilane (TMS).

3.6 Multiplicity of Lines and the Coupling Constant $\mathcal{J}$

When observing a high-resolution spectrum one can often observe a *multiplicity of lines* very close to one another. CH_3, CH_2 and OH protons for example are formed by a multiplicity of signals. (See figure 3.2)

The appearance of these multiple signals is interpreted as an indication of a magnetic interaction among protons of groups. This interaction is generally defined as *nuclear coupling*. The intensity of this coupling is defined by the *coupling constant* $\mathcal{J}$, and is measured in Hz.

The appearance of a spectrum is a function of the relative difference between the value of $\mathcal{J}$ with respect to the chemical shift separation, $\Delta\delta$, between the coupled nuclei. When $\mathcal{J} \ll \Delta\delta$, then the spectrum is called a 'first order spectrum' and the number of lines in which a signal is split is related to the number of coupled equivalent nuclei by the equation

$$x = 2nI + 1$$

where x is the number of lines of the multiplet, n is the number of coupled equivalent nuclei and I is the nuclear spin number. The intensities of the lines can be predicted similar on the basis of the Pascal triangle.

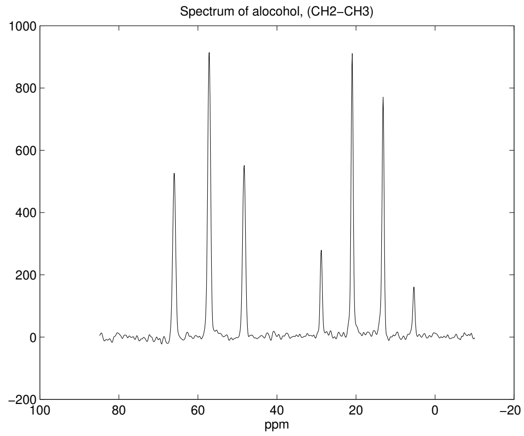


Figure 3.2: Spectrum of alcohol showing a 1-2-1 CH_2-group and a 1-3-3-1 CH_3-group

Unlike the chemical shift, $\mathcal{J}$ is independent of the field strength temperature, solvent and concentration of a sample.(See [1] for more details)

3.7 Various kinds of NMR-Spectra

The following sections will give a brief overview of the kinds of spectra used in the clinical environment. ^{1}H , ^{13}C and ^{31}P are the most common.

3.7.1 ^{1}H NMR

Because of its wide spread distribution in the compounds and its high nuclear sensitivity ^{1}H was the first nucleus to be observed with NMR. ^{1}H also has a natural abundance of almost 100% which means that, in principle, every proton on earth can be examined with NMR.

The range of compounds is about 10 ppm, which can be quite inconvenient if the chemical environment contains many compounds, resulting in a complex spectrum which is difficult to analyze because of the high degree of overlapping lines.

The complexity of the ^{1}H spectrum is one of the main reasons for developing NMR instruments with higher magnetic fields. A higher magnetic field will result in a broader spectral width and will make almost all spectra first order spectra.

Common metabolites examined in ^{1}H spectra are for example lactate, creatine, choline, acetylasparate and lipids.

3.7.2 ^{13}C NMR

The natural abundance of ^{13}C is very small and the relative sensitivity is about 60 times smaller then ^{1}H . It is therefore necessary that the samples taken are 60 times more concentrated then the ones used for ^{1}H , which is most of the times impossible. There are several ways to overcome this problem:

1. Enriching the molecules to be studied with ^{13}C . This is very expensive and sometimes impossible.
2. Taking advantage of the pulse technique, one can proceed to acquire several FIDs until an acceptable SNR is achieved.

Because of the fact that the possibility of spin-spin interaction between two ^{13}C nuclei is very small, the intensity of these kind of signals is very small. It requires special pulse sequences to detect these signals. A better sensitivity can be obtained by proton decoupling.

The range of a ^{13}C spectrum is about 220 ppm with respect to TMS at a B_0 field of 4.7T.

3.7.3 ^{31}P NMR

^{31}P NMR has a relevant meaning for biological applications. It's relative sensitivity with respect to ^{1}H is about 6.6×10^{-2}. The natural abundance of ^{31}P is 100% and can thus be observed without any major difficulties.

A ^{31}P spectrum is easy to identify in spectra obtained from biological specimens.

An interesting feature about ^{31}P resonance lines of compounds bearing acid/base dissociable groups is the chemical shift dependence from the pH, which provides a unique possibility to determine the inter cellular pH in intact cells or organs.

3.8 *In Vitro* versus *In Vivo* measurements

Without going into details, it's not difficult to understand that *in vivo* measurements are of less quality then *in vitro* measurements. Mostly this is due to the fact that measurements on living organs require organs to lay still, for example for the heart and lungs this raises problems, resulting in spectra of poor quality.

Another problem is the influence of the environment of the nucleus observed. Living objects or humans have a continuous sequence of reactions within their bodies, which of course influences the measurements taken, more then in *in vitro* measurements. (See [6] for various *in vivo* and *in vitro* experiments.)

3.9 The Processing of Spectroscopic Data

From the previous sections it may be clear that processing of spectroscopic data is not as simple as it may seem. In order to obtain spectra that can be analyzed various methods and techniques have to be applied. In all cases the validity of all these methods has to be guaranteed.

Since NMR spectra become more and more important in characterizing certain diseases, new methods are applied. One of these methods is called Spectroscopic Imaging (SI). In this way spectra are take over a wide area, this can easily lead to enormous amounts of data. For example, a SI of 40x40 voxels already contains 1600 spectra, which is for humans impossible to analyze interactively.

There is a need for automatic spectrum analyzers, that can handle these large amounts of data, are capable of characterizing individual spectra and are able to present the data.

3.10 References

[1] Alberto Spisni
1D Spectrum Analysis
Magnetic Resonance Spectroscopy in Biology and Medicine, p31-p58 (1992)

[2] Wim Bovée and Joris Creyghton
Acquisition and Processing of the NMR Data
Magnetic Resonance Spectroscopy in Biology and Medicine, p133-p148 (1992)

[3] E. Breitmaier, W. Voelter
^{13}C NMR Spectroscopy
Verlag Chemie 1974

[4] Frank A. Bovey
High Resolution NMR of Macromolecules
Academic Press, New York and London 1972

[5] Richard R. Ernst, Geoffrey Bodenhausen, Alexander Wokaun
Principles of Nuclear Magnetic Resonance in One and Two Dimensions
Oxford university Press 1990

[6] J.D. Certaines, W.M.M.J. Bovée, F. Podo
Magnetic Resonance Spectroscopy in Biology and Medicine
Pergamon Press, 1992

Chapter 4

Artificial Neural Networks

4.1 Introduction

The roots of neural networks date from the 19-th century. The neurologist Jackson (1869-1958) was the first man with ideas about distributed, multilevel conceptions of processing systems in the human mind. Since that period of time neurologists and psychologists are working on models of the functioning of the human brain.

Several people contributed to a concept of a model that appears as an *artificial neural network*.[1] Two of the most important early day scientists were Hebb and Rosenblatt.

Hebb's ideas about synaptic modification and Rosenblatt's vision of the human information processing system as a dynamic, interactive self-organizing system are the core of neural networks. There are many other important scientists that build on the ideas of Hebb and Rosenblatt, which has let to a series of different models.

Although none of the models is capable of simulating the exact human behavior, some models come very close in simulating a certain part of it. For example, some of the problems neural networks appear to be useful for are:

- pattern completion
- pattern classification
- pattern recognition
- filtering
- feature detection

It is beyond the scope of this thesis to give a full description of all types of neural networks that exist and their specific characteristics. In order to get

[1] Since the neural networks discussed in this chapter will all be artificial, the term *neural network* will be used instead of *artificial neural network*.

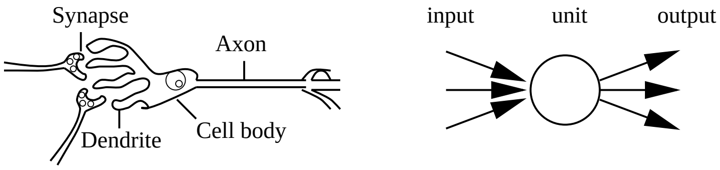


Figure 4.1: Physical structure of a neuron and it's simplified representation in an artificial neural network

an idea about the different kind of networks available I refer to the PDP-research group.([2], [3])

Since this thesis is about classification and detection of special features of NMR-data, most attention is focussed on the model that is most commonly used for these kinds of problems: the *feed forward neural network*.

4.2 A Mathematical Model

As mentioned in the introduction an artificial neural network is a simple model of the biological information processing system. Figure 4.1 gives the representation of a neuron and it's simplified model called a unit. Like a neuron, a unit can have multiple inputs, as indicated by the arrows on the left side of the unit, and it has one output, which can be connected to various other units.

A neural network is a collection of units which are connected with each other, some of the units get stimuli from the external environment. In physical neural networks this can be sound, vision or other stimuli, others represent the output, in the form of a certain action. Not all of the units have to be part of input or output units, there can also be units that represent some internal state of a neural network. Figure 4.2 gives an example of a four layered fully connected neural network.

4.2.1 Description of the Components of a Neural Network

Following Rumelhart and McClelland[2] their are eight major components of a neural network:

- A *set of processing units*
- A *state of activation*
- An *output function* for each unit

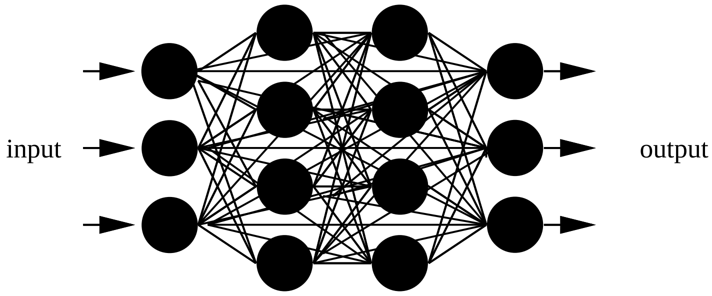


Figure 4.2: Example of the architecture of a four layered fully connected artificial neural network. There must be an input- and output for the system.

- A *pattern of connectivity*
- A *propagation rule* for propagating patterns of activities through the network of connectivities
- An *activation rule* for combining the inputs impinging on a unit with the current state of that unit to produce a new level of activation for the unit
- A *learning rule* whereby patterns of connectivity are modified by experience
- An *environment* within which the system must operate

Each of these components will be shortly described.

Set of processing units

The first step in building a neural network is specifying a set of processing units, and defining their representation. In some models these units may represent particular objects such as features, in others they may be abstract elements over which meaningful patterns can be defined, for example letters within words, line elements within letters etc.

The unit's job is very simple, it receives inputs, and computes an output as a function of the inputs.

Rumelhart and McClelland distinguish three kinds of units:

input units the set of units that receive input from external sources, like for example sensory inputs

output units the set of units that send signals to the outside of the system

hidden units the set of units that are not visible to the outside of the system, and represent the internal state of the system, the aim of these units is to provide an internal coding mechanism for problems that can not be solved using only in- and output units, e.g. a two layered neural network

Let N be the total number of processing units in the system, then the i-th unit will be designated u_i.

State of Activation

In order to keep track of the activation development within the system we need a representation of the state of the system at time t. Since all units need a state there are N states that describe the system. The state of a unit u_i at time t will be denoted by $a_i(t)$, a_i is state of activation or simply activation.

Different models make different assumptions about the activation values a unit may take on. Activation values can be discreet or continuous. If they're continuous they may be bounded, for example within the interval $[0, 1]$, or unbounded. In case they're discreet, they may take on a small finite set of values, for example $\{0, 1\}$, or $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Each of these assumptions leads to a model with different characteristics.

Output Function of a Unit

Units interact by transmitting signals to their neighbors. The degree to which they effect their neighbors is determined by the degree of activation of a unit. In other words, there is need for a function:

$$o_i(t) = f(a_i(t))$$

which maps the current state of activation $a_i(t)$ to an output signal $o_i(t)$.

The function f usually has some sort of threshold Θ or bias value, which is individually set or learned for each unit. It can be continuous (e.g. linear or some sigmoid function) or discreet (e.g. 1 if $f > \Theta$ else 0).

Connectivity within a network

Units are connected with one another. It is this pattern of connectivity that constitutes what the system "knows" and how it will respond to any arbitrary input.

The pattern of connectivity is represented by a matrix W in which entry w_{ij} represents the strength of the connection from unit u_j to unit u_i. W is called the *weight matrix*. If the weight $w_{ij} > 0$ then unit u_j is said to *excite* unit u_i, if $w_{ij} = 0$ then there is no direct connection between unit u_j and unit u_i, and if $w_{ij} < 0$ then unit u_j is said to *inhibit* unit u_i. The strength of a connection is defined by the absolute value of w_{ij}.

In the general case, units can receive different kinds of inputs which must be summed separately, leading to several weight matrices. An example is a matrix for the inhibitory connections and one for excitatory connections. The aim of these different weight matrices is to provide different types of connections in the present sense.

Rule of Propagation

The rule of propagation takes the output vector $o(t)$ and combines it with the connectivity matrix to produce a *net input* for each type of input.

For reasons of simplicity we restrict ourselves to only one type of connection and write net_j for the net input of unit u_j.

The rule of propagation is in its most simple form a multiplication of the weight matrix $W(t)$ with the output vector $o(t)$ or in terms of a simple unit the net input for unit u_i is determined as follows:

$$net_i = \sum_{j}^{N} w_{ij} o_j$$

where net_i is the net input for unit u_i, w_{ij} is the weight of the connection between unit u_j and unit u_i, and o_j is the output of unit u_j.

Activation Rule

The activation rule is used to combine the net inputs of each type and the current state of a unit to produce a new state of activation. This activation rule is denoted by the function F.

F takes as its arguments the vector $a(t)$ of activation values at time t and the *net* vector of net inputs and computes the new state of activation. F with only one type of net input can be written as follows:

$$a_i(t+1) = F(a_i(t), net_i(t)),$$

The function F is usually the identity function, so that:

$$a_i(t+1) = net_i(t)$$

The Learning Rule

Learning rules in neural networks require changing the strengths of connections between units, e.g. the weights, resulting in connection or disconnection of units, changing output functions and activation rules. There are many learning rules available for various models, but most of the learning rules are variants of the Hebbian learning rule.

The basic idea behind Hebb's rule is that when unit u_i receives input from unit u_j, and both are highly active, then the weight w_{ij} should be strengthened. In a general form this can be stated as follows:

$$\Delta w_{ij} = g(a_i(t), t_i(t))h(o_j(t), w_{ij}),$$

where $t_i(t)$ is a sort of teaching input to unit u_i, $g()$ and $h()$ can be any function. In the most simple form the functions $g()$ and $h()$ are proportional to their first argument, resulting in:

$$\Delta w_{ij} = \eta a_i o_j,$$

where η represents the constant of proportionality called the *learning rate*.

There exist more learning rules, some of them will be explained in the section of the *feed forward neural network*, for others, not relevant for this thesis, I refer to the literature.

Representation of the environment

The most common representation for neural networks is a time-varying stochastic function over the space of input patterns. This means that at any point in time, there is some probability that any of the possible set of input patterns is impinging on the input units. Figure 4.3 shows a general environment for neural networks.

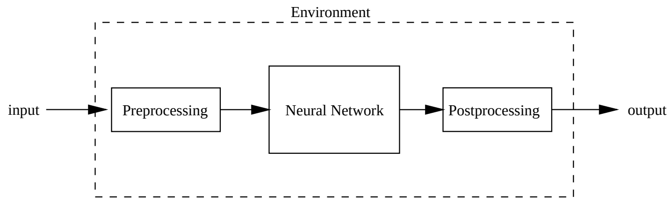


Figure 4.3: General environment for neural networks. Note that the pre- and postprocessing may be dummy procedures.

The most simple form of neural networks is one in which there is a stable probability distribution over the set of input patterns which is independent of the past inputs and past responses of the system.

4.3 Learning in General

Solving any problem using neural networks includes finding an optimal weight matrix that solves the problem.

Rumelhart en McClelland have shown that a set of orthogonal input patterns and linear independent input patterns can be learned ideally.

In practice however, we hardly ever have situations in which the input patterns are orthogonal or linear independent to each other. In these cases we need to find a method that either generates an orthogonal or linear set of input patterns from the data we have, or need to apply special kinds of algorithms that keep the error as low as possible.

4.3.1 Problems with Learning

The most common problems when learning a neural network are:

- high *oscillation of the weights*, resulting in a system that doesn't converge to a minimal error
- the appearance of *local minima* in the error-surface. There the learning of the network stops.
- *over fitting* of the neural network, meaning that the network is not capable in generalizing, e.g. the results on a test-set will be unpredictable

These problems can be dealt with, depending on the methods used. They will be described for the feed forward neural network in the next section.

4.4 The Feed Forward Neural Network

If the similarity structure of the in- and output patterns are very different, a two layered neural network is not capable of performing the required mapping from input to output units. In this case the system needs a sub layer that provides an internal representation, or recoding, of the input patterns. This layer is called the *hidden layer*. If a hidden layer is used, there is always a recoding of the input patterns in the hidden units in which the similarity of the patterns among the hidden units can support any required mapping from the input to the output units. This means that if the right connections from the input units to a large enough set of hidden units are present, we can always find a representation that will perform any mapping from input to output through these hidden units.

Feed forward neural networks can contain one or more hidden layers, fig 4.4 shows an example of a three layered ff-network. However, any mapping in principle can be achieved when using one hidden layer with enough hidden

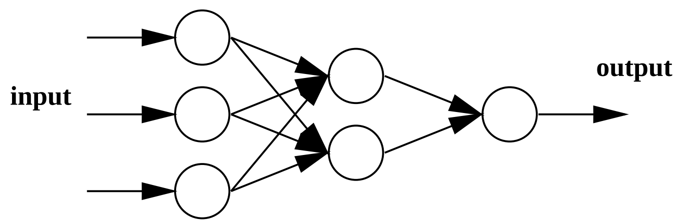


Figure 4.4: Example of a three layered ff-network

units. It is however sometimes more convenient to use more then three layers.

4.4.1 Processing of Information

The basic idea of feed forward networks is that each layer, which can be the input or any hidden layer, sends its output to the next layer and receives its input from the previous layer. The output layer only receives input from the previous layer. In the case of the input layer this means that it receives input from the environment, and the output layer sends its output to the environment.

Given an input pattern, the output pattern is computed by a forward pass which computes the activity levels of each layer in turn using the already computed activity levels in the earlier layers.

Since any mapping done by a linear feed forward network can be done with a network using two layers, there is no point in using the linear activation function for feed forward networks with more then two layers. In order to make advantage of the hidden layers, a nonlinear activation function has to be used. There are however two constraints for this function for reasons that will become clear in the next section.

1. it must be differentiable
2. it must be nondecreasing

4.4.2 The Delta Rule for Feed Forward Networks

The learning algorithms of *supervised learning* are usually based on the *delta rule*. The delta rule is essentially an implementation of gradient descent in sum-squared error for linear activation functions. In the linear case, without hidden units, the error surface is shaped like a bowl and the gradient descent is guaranteed to find the best set of weights.

This delta rule requires an input- output pair p and computes the error signal δ, using the following formula:

$$\Delta_p w_{ji} = \eta(t_{pj} - 0_{pj}) i_{pi} = \eta \delta_{pj} i_{pi},$$

where $\Delta_p w_{ji}$ is the change in weight from unit i to unit j, η is the constant of proportionality, or learning rate, $\delta_{pj} = (t_{pj} - 0_{pj})$ is the error signal, t_{pj} represents the target output for j-th element of target pattern t, 0_{pj} is the j-th element of computed output pattern o, and i_{pi} is the i-the element of input pattern i. For the derivation of this formula see [2].

Unfortunately, the delta rule in this form can not be applied to the feed forward networks with more then two layers. If hidden units are used, it is not so obvious how to compute the derivatives, and the error surface may not be shaped as nicely as in the linear case. It is very well possible that the error surface contains holes, meaning that the system can get stuck in local minima.

For the case of nonlinear activation functions we need to adjust the delta rule to a more general case. This generalized delta rule only works with nonlinear differentiable and nondecreasing activation functions, threshold activation functions do not satisfy these constraints because their derivative is infinite at the threshold, and zero elsewhere. The generalized delta rule takes the following form:

$$\Delta_p w_{ji} = \eta \delta_{pj} o_{pi},$$

where δ_{pj} looks like:

$$\delta_{pj} = (t_{pj} - o_{pj}) f_j'(net_{pj}),$$

if unit u_j is an output unit, and

$$\delta_{pj} = f_j'(net_{pj}) \sum_k \delta_{pk} w_{kj},$$

if unit u_j is not an output unit, e.g. a hidden unit. For a derivation of this formula, though not a formal, see [2]. As can be seen from this latest formula, the weight changes are computed by a recursive process, if the unit in question is not an output unit. In this case all weights get updated according to the set of processing units they belong to. The application of the delta rule takes two phases.

Phase one is the *forward-propagation* of the input through the network to compute the output value o_{pj} for each unit. The output is then compared with its target value, and the error signal δ is computed.

Phase two is the so called *back-propagation* phase, first the error signal δ is computed for each of the output units, then the weight changes are

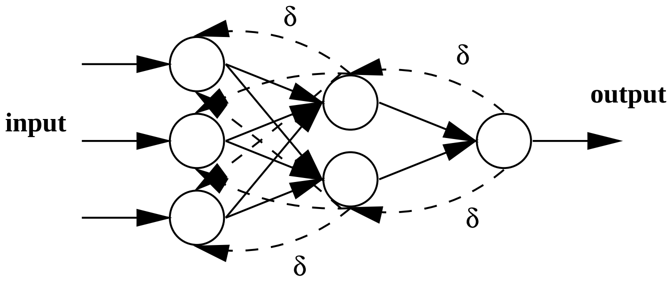


Figure 4.5: Example of a three layered ff-network with back-propagation, the δ term represents the error signal that is propagated in the backward pass

computed for all weights that connect any hidden unit to an output unit. The next step is to compute all δ terms for the hidden units and apply the weight changes according to the rule for hidden units. In this way the error is back-propagated one layer, and this can be repeated for any layer.

Figure 4.5 gives an example of the two passes in the ff-network using the back-propagation rule.

4.4.3 Activation Functions

As mentioned in the sections above the activation function of a feed forward network must be differentiable and nondecreasing. The functions used usually have the shape of a sigmoid function. A possible activation function could be:

$$o_{pj} = \frac{1}{1 + e^{-(\sum_i w_{ji} o_{pi} + \Theta_j)}},$$

where Θ_j is a *bias* similar in function to a threshold. Assumed here is that the output $o_{pj} = a_{pj}$.

To use this activation function one has to compute its derivative with respect to the net input, then the δ term can be computed, and the weights can be changed accordingly.

There are a number of activation functions that can be used with ff-networks using back-propagation, each of them depends on the output required. The most important activation functions are listed below.

- Log-sigmoid, this function has output values between 0 and 1
- Tan-sigmoid, this function has output values between -1 and $+1$
- Linear, with values in the range $(-\infty, +\infty)$

Each of these functions can be used for any special purpose. There are however more activation functions, for a list see [7].

4.4.4 Learn Parameters

Learning a neural network using back-propagation is a process of changing weights in such a manner that the error function reaches it's global minimum. For the proof of the existence of an error function and an error-surface see [5].

Many error-surfaces, however, have flat areas, and steep holes, e.g. local minima, which means that learning does not take place without any problems. Figure 4.6 gives an example of a possible surface and the traject in weight space due to training.

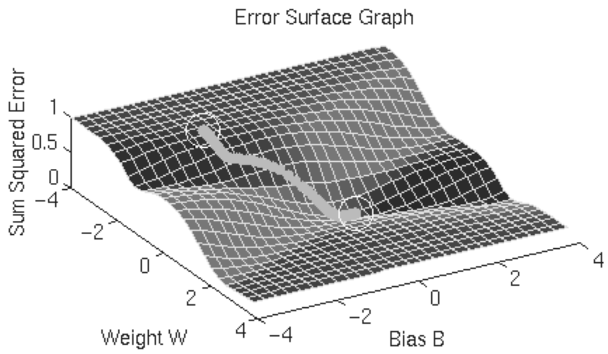


Figure 4.6: Example of an error-surface and the traject to find an optimal set of weights and bias terms to solve the problem. The error-surface depicted here is one of a single layer neural network, with one weight W and one bias B, using the back-propagation rule.

There are a number of parameters that can speed up the learning process and overcome some of the problems. They will be described briefly in the following text.

Learning Rate

The *learning rate*, or *constant of proportionality* affects the weight changes. To make the learning process as fast as possible the learning rate is chosen as large as possible as long as the "weights" do not yet start to oscillate.

Momentum

The momentum term has two effects, it speeds up the learning process and it can overcome the problem of steep holes. The weight change with momentum term looks as follows:

$$\Delta_p w_{ji}(n+1) = \eta(\delta_{pj} o_{pi}) + \alpha \Delta_p w_{ji}(n),$$

where η is the learning rate, and α represents the momentum term and n represents the n-th presentation.

As can be seen from this formula α determines the effect of past weight changes on the current direction of movement in weight space. The effect of using a momentum term is that it filters out the high-frequency variations of the error-surface in weight space. The use of the momentum enable the increase of the effective weight steps, a property that allows 'jumping' over steep holes.

The values for a momentum are usually about 0.9, this is taken from experience rather then from formal analyses.

Squared-Sum Error Goal

The learning rate and momentum term speed up the learning process, but do not effect the quality of learning. To specify how well a neural network must learn a mapping from input to output, an overall *squared-sum error goal* is introduced. The total squared-sum error E is defined by:

$$E = \sum_p E_p = \sum_p \sum_i (t_{pi} - o_{pi})^2,$$

where $E_p = \sum_i (t_{pi} - o_{pi})^2$ is the error over input- output pair p.

If training over all patterns has reached the squared-sum error goal training will stop and the testing of the neural network can start.

4.4.5 The Influence of Biases

The use of biases gives a network more degrees of freedom to learn a desired function. They increase the chances that a back-propagation network will find an acceptable solution for the mapping and in many cases it decreases the number of epochs required to train the network.

Figure 4.7 shows the effect of using biases, the function shown here is the tan-sigmoid, which has output values between $+1$ and -1. Applying a bias makes the activation function shift with a term θ which can be seen as a sort of threshold, if the net input is bigger then a certain value θ, it will count significantly otherwise the influence of that unit will be rather small. The biases are usually learned per neuron in a simular way as the weights.

4.4.6 Choosing the Number of Hidden Units and Hidden Layers

One of the major problems of defining a neural network architecture is how many units should be used for the hidden layer. Usually this is determined

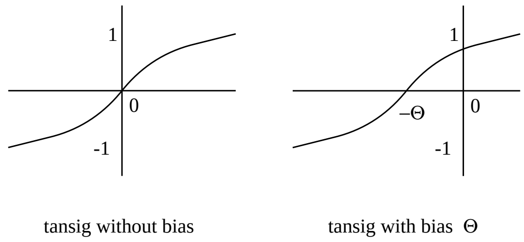


Figure 4.7: The influence of biases on the activation function, it can be seen from these two pictures that the bias Θ behaves like a sort of threshold function.

by experimentation, where the correct percentage of input- output pattern pair is set out to the number of hidden units. In this case one can determine the number of hidden units that's capable of solving the problem. However, when dealing with a very complex problem such a method takes too much time and can not be used. There are several other approaches to estimate the number of hidden units, but none of them are formally safe to use. (See [7] for more information)

In general it can be said that the number of hidden units represents the complexity of the problem and vice versa, e.g. a complex problem requires many hidden units.

As mentioned before all problems that can be solved using a neural network with more then three layers can be solved by a network with just three layers[2], however for some problems it is easier and less time consuming to use more then three layers, since the problem to be solved would have many hidden units in the hidden layer, it would take a great deal of time to train such a network.

4.5 Representation of In- and Output Data

Another important aspect of neural networks is how we are going to present our data to the network. In what form, how should we represent our input and what output do we want?

In the theoretical case the feed forward neural network should be able to learn almost any representation, under the condition that there exists a classification mechanism.

In practice we are dealing with computers with certain limits that may not be adequate to solve the problem in an acceptable amount of time, in this case we need to do some pre-processing of the actual data to fit it in a representation the neural network, and the computer can handle.

There are a number of methods that can be used, if we have too much data, like for example in NMR-data (about 32K data points per FID) it is sometimes necessary to reduce the data to an acceptable size, e.g. by filtering out those coefficients that are representative for the information contents.

Other methods for presentation of the data are usually in the form of normalizing, standardizing and rescaling the data. Whether or not these methods are to be applied depends on the data you have.

4.6 Solving Problems using Neural Networks

To solve any problem using neural network, the problem has to be solvable, e.g. there has to exist a mapping from input to output, in the first case, and has to be a kind of problem that can be handled by neural networks.

To find a solution for the problem a number of steps have to be taken:

1. find a neural network model that can handle the problem
2. define the architecture of the network you're going to use, e.g. how many layers, how many units, the connectivity of the network etc.
3. pick a possible activation rule, this depends of course heavily on the architecture of the network
4. find suitable values for the weights, biases, learning rate and momentum, if they're used
5. determine the error-goal
6. train the network using a representative dataset
7. test the network using a representative dataset
8. verify the results

Steps 2 to 8 can of course be repeated until satisfying results are achieved.

4.7 References

[1] David E. Rumelhart, James L. McClelland and the PDP research group
Parallel Distributed Processing
Explorations in the Micro structure of Cognition
Volume 1: Foundations
1986 , The Massachusetts Institute of Technology

[2] David E. Rumelhart, James L. McClelland and the PDP research group
Parallel Distributed Processing

Explorations in the Micro structure of Cognition
Volume 2: Psychological and Biological Models
1986 , The Massachusetts Institute of Technology

[3] James L. McClelland and D. E. Rumelhart
Explorations in Parallel Distributed Processing
A Handbook of Models, Programs, and Exercises
1988 , The Massachusetts Institute of Technology

[4] Dana Z. Anderson
Neural Information Processing Systems
American Institute of Physics, 1988

[5] Robert Hecht-Nielsen
Neurocomputing
Addison-Wesley Publishing Company, Inc. 1990

[6] Howard Demuth, Mark Beale
Neural Network Toolbox User's Guide
The MathWorks, Inc. 1993

[7] Warren S. Sarle
Neural Networks FAQ
URL: "ftp://ftp.sas.com/pub/neural/FAQ.html"

Chapter 5

Spectral Classification of Ovarian Cyst Fluids

5.1 Introduction

Most ovarian neoplasms occur as cystic tumors. Once a diagnosis points to the presence of a tumor, the next question will be whether this tumor is benign or malignant.

To answer this question a project is performed at the University Hospital Nijmegen St. Radboud by Vierzen et al. (See [1]) The aim of this investigation is to:

- analyze the range of NMR visible metabolic components in cyst fluids
- find out whether differentiation between benign and malignant ovarian cysts can be made on the basis of their metabolic composition

They used a standardized method, as described in [2], to gather and analyze the results.

5.2 The Problem Domain

The spectra obtained from these cyst fluids are complex and contain many overlapping lines, figure 5.1 shows an example of such a spectrum.

The analysis of such a spectrum requires a considerable amount of time. The elementary question is, whether this analysis can be automated, and how this process can be integrated into the clinical environment.

One of the methods that may be used for analysis is the application of artificial neural networks. Research done with neural networks, especially

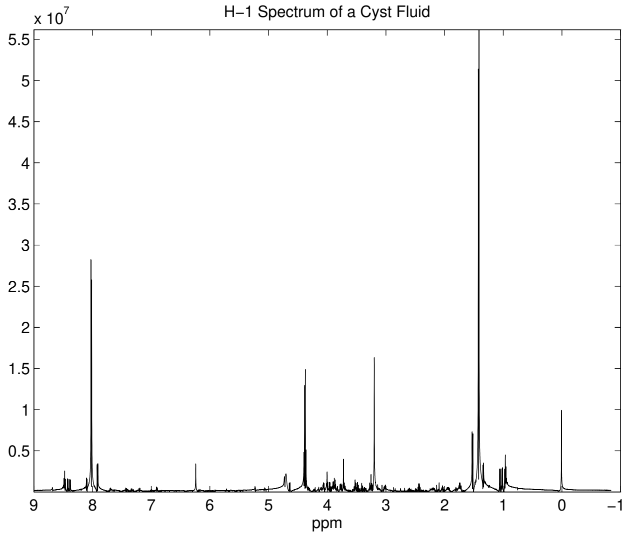


Figure 5.1: Example of a high-resolution ^{1}H spectrum of a cyst fluid

in the field of chemistry, have shown that neural networks in combination with some preprocessing tool, like for example Principle Component Analysis (PCA), give promising results for the analysis of spectra. It's our believe that these neural networks can also be used in combination with high-resolution ^{1}H spectra, whether preprocessing will be required or not remains an open question, which we hope to solve in this experiment.

5.3 Materials and Methods

Samples of 24 ovarian cysts were obtained by aspiration during operation of a patient with a tumor. All samples were stored at $-35^{o}C$.

Before NMR analysis all samples were passed through centrifugal ultra filters. The ^{1}H spectra were than obtained at a 600 MHz Bruker WH600 spectrometer [2]. Trimethylsilyl propanic acid was used as an internal standard for chemical shift calibration and the estimation of metabolite levels.

5.4 Classification of Total Spectra

Our first attempt was based on classification of the cyst spectra using a table containing metabolite levels of the various compounds found in the spectra (See [1]). The table was made up of nine classes, each of these classes represented a certain tumor: benign or malignant. The aim of this experiment is to show that:

artificial neural networks can be used to classify an unknown set of cyst fluid spectra, based on earlier experiences with a well known set of cyst spectra

After a learning phase the network should be able to classify the training set, and also classify an unknown set of cyst fluid spectra, called the test set.

Due to the low number of spectra (24) and the fact that certain classes only contained one spectrum, we had to split the dataset into three classes, each containing four spectra of the same class.

The network was trained using three spectra out of each class, resulting in a training set of nine spectra. The test set was created by the remaining spectra. This method is called the *leave one out* method.

In order to get some idea of the performance of the network, this process was repeated four times, each time using a different training set and a different test set. The advantage of repeating this process four times is that each spectrum of each class is used as a part of the test set, in this case the overall performance can be computed by taking the average over all four sessions.

5.4.1 Choosing the Network

The networks most commonly used for pattern recognition are the feed forward neural networks combined with the back-propagation learning paradigm. These are the kind of networks we first focussed on.

Since we're using the Matlab Neural Network Toolbox we're restricted in using three layers, one input, one hidden and one output layer. As mentioned in the chapter about neural networks theoretically three layers should be enough to solve the problem.

The Input Layer

Since the spectra from the table contained 47 compounds with values in the range from 0.0 to 500.0, it seems trivial to use 47 input units for the input layer. Each of these compounds was represented by a floating point number, the input units therefor are also represented as floating point numbers. Since the table was made up of metabolite levels only, no information about the position of the metabolites was stored.

The spectral components were already normalized with respect to TMS, so no further normalization was required.

The Hidden Layer

Due to the fact that there is no adequate rule of thumb to determine the number of hidden units that should be used, the number of units required was determined by experiment.

The network was trained using all 9 input spectra and the number of hidden units was taken variable, in the range from 20 to 80 units in steps of 5 units. In order to find a number of hidden units that could solve the problem, an error plot was made for each number of hidden units.

Whether or not a network reaches it's error goal within the maximum number of epochs depends on where the network starts in weight space and whether or not it is capable to overcome the local minima it encounters. This in turn can be influenced by some parameters, which are described below.

The Output Layer

Since we have three classes, the output layer contains three output units, one for each class. If the input spectrum is of class 1 then the output unit indicating class 1 should be turned on, the others should stay off.

5.4.2 The Activation Rule

The Matlab Neural Network Toolbox offers two important activation rules for learning back-propagation networks. The first is the log-sigmoid, with output values in the range of [0,1]. The second is the tan-sigmoid, with output values in the range of [-1,+1]. After learning with both activation rules it turned out that learning with the log-sigmoid activation rule was much faster than learning with the tan-sigmoid function. It was mainly for this reason we focussed on the log-sigmoid function.

5.4.3 Parameters

There are some parameters that influence the performance of the network. In our experiment these are:

- the learning rate
- the momentum term
- the squared sum error goal
- the initial values for the weights and biases

Learning Rate and Momentum

After several training sessions and a bit of experimenting, it showed that a momentum of 0.9 and a learning rate of 0.05 performed reasonably well, e.g. the error was continuously decreasing and the speed of learning was relatively fast. Increasing the learning rate and/or momentum term lead to heavy oscillating weights which means that the network is not capable of reaching its error goal, and can not be "learned".

It must be said, that even with these apparently "optimal" parameter values, the network didn't always reach its error goal, within the maximum number of epochs allowed.

The Squared Sum Error Goal

In order to determine the squared sum error goal we used the method as described by Sarle [1], the so called *stopped training* method. This method is based on training the network with a lot of hidden units and following the squared sum error until the error starts increasing again. After repeating this process several times, a sort of minimum error goal can be found, this however doesn't have to be the global minimum error, which remains to be an inherent problem of this neural network approach.

The error goal was determined on 0.01 in this experiment, this was not the minimal error found. Since only a few experiments reached the minimal error, we had to make an assumption about the relevant value. This value was set to 0.01. The gain beyond the determined error-goal was for almost all cases less than 10%.

Initial Weights and Biases

After several experiments it turned out that the network reached its best performance when the initial weights and biases were randomly distributed over the interval [-0.05, +0.05]. In most cases the error goal was reached and the network was capable of overcoming the local minima.

It also turned out that the starting point in weight space is a very important factor in whether or not the network is capable of learning a certain mapping. With large initial weights, the chances of the network reaching its error-goal were much less then with smaller initial weights. Also the speed of learning depends heavily on these initial weights. However, choosing proper initial weights doesn't always result in a network that can be learned.

5.4.4 Results

The network was trained with a different number of hidden units per training session ranging from 20 to 80 hidden units in steps of 5 units, each session was repeated four times to get a better idea about the classification mechanism.

The test set was made up off one spectrum from each class. When using four training sessions with a fixed number of hidden units, it means that each spectrum from each class is exactly once part of the test set in each training session.

In most cases the error-goal was reached before 20000 epochs, which was to be the maximum number of epochs.

After training the network was tested using the same dataset as it was trained with. The results of this test were very close to 1, meaning that the network was capable to correctly classify the data it was trained with. Figure 5.2 shows the average classification values over four training sessions plotted against the number of hidden units.

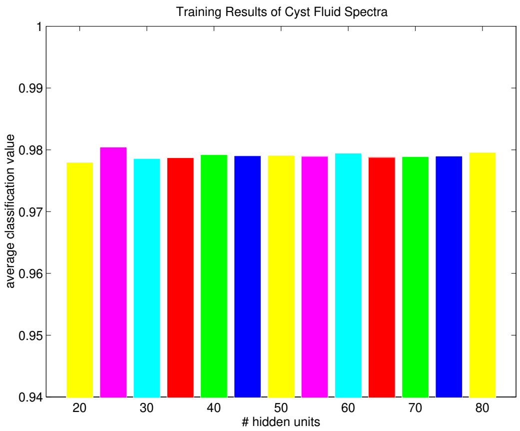


Figure 5.2: The average classification values over four training sessions are plotted against the number of hidden units.

The output values for all spectra in the training set were in the range of 0.85 to 0.99, which is quit good actually.

Training became faster by increasing the number of hidden units. Figure 5.3 shows the number of epochs necessary to reach the error-goal and the error-curve for some of the training sessions. This figure also illustrates several intersections of the error surface. The figures 5.3(a) and 5.3(b) show a slowly decaying curve with a bending point before 2000 epochs, which vanishes when more hidden units are used. When we look at figure 5.3(a) we see that this slowly decaying curve is only present in two training sessions, this indicates that it very much depends on the starting position in weight space how the error curve will evolve and if it will reach its error-goal.

Whether or not training was successful was determined by the output value, given a specific spectrum. If the output value indicated the correct class (one of three), and it's value was greater than 0.75, then training was said to be successful. Applying these values to the values presented by figure 5.2, this means we have a 100% result for the training set.

After training the network was tested using the three remaining spectra. For testing the same values were used to determine whether or not a spectrum could be classified, e.g. a value greater than 0.75, and indicating the correct class, means that the spectrum was successfully classified. This lead to the following results (See figure 5.4):

When comparing the training results with the test results, it can be said that the network was not very successful in classifying unknown spectra, although the training results were excellent. One of the main reasons for this phenomenon is that the number of spectra used to train the network was to small, which means that the network was not capable to generalize the spectral classes in an adequate way. This can be due to the fact that the network did not learn enough details about the spectral classes, or it learned too much, which means that the spectra were to wide spread in hyper space, and the intersections between classes were not empty.

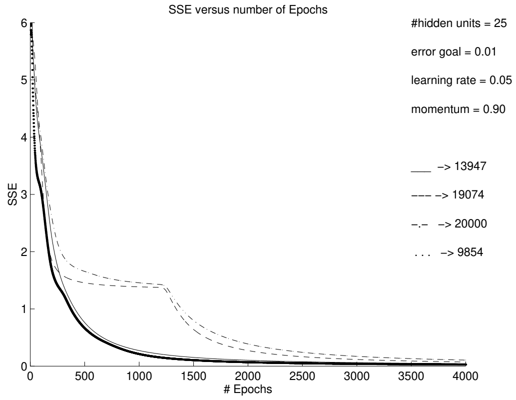
Figure 5.4 also shows that increasing the number of hidden units does not necessarily lead to better performance of the network. The best performance was reached using 35 hidden units.

5.4.5 Conclusions

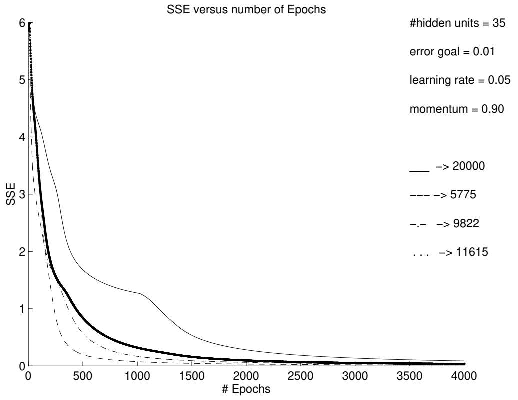
Although it is known from literature, this experiment indicates again that to train neural networks with many input units, a lot of training data are needed.

Considering the complexity of the data, and the number of spectra used we can say that differentiation between different classes is not a straight forward process, the results would otherwise have been more promising. When the difference between the classes would have been clear the neural network should have been able to classify the input data to the correct class, however the differences between the classes appear to be very small.

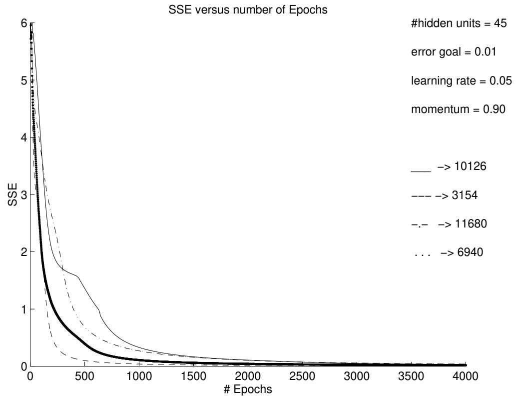
Especially when these kind of methods have to be used in a clinical environment the validity of these methods play a very important role, therefor it is not possible to make a statement about the application of neural networks at this stage.



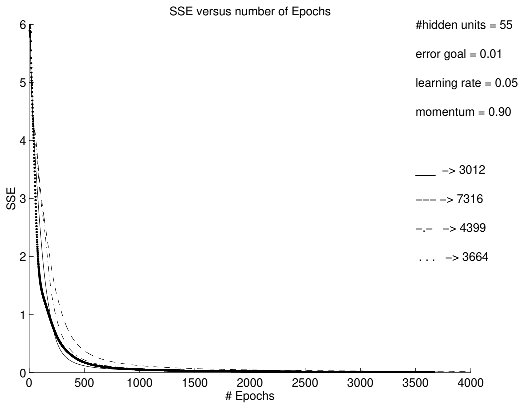
(a) SSE with 25 hidden units



(b) SSE with 35 hidden units



(c) SSE with 45 hidden units



(d) SSE with 55 hidden units

Figure 5.3: Number of epochs used to reach the error goal for various hidden units and various training sessions. (SSE = Squared Sum Error) The numbers in the agenda of the figures represent the number of epochs for each trial, which are marked by the various line-shapes, e.g. dotted lines etc.

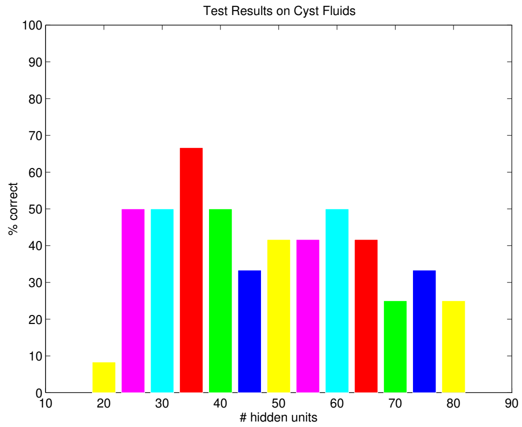


Figure 5.4: Plot of the test results, the values given are the average values over four test sessions. Increasing the number of hidden units does not necessarily lead to better performance of the network.

Another important factor is the tuning of the network, since there are no predefined rules for constructing a neural network, it's very difficult and time consuming to find a certain set of parameter values that solves the problem.

Because of the relative small datasets, it's hard to say whether or not the network learned exactly what it was supposed to learn. When focusing on the training data, one could easily draw the conclusion that the network learned an exact match of the classes involved, e.g. each spectrum in the training set was assigned to the correct class, but after testing it showed that the things the network learned were not the things it was supposed to learn.

5.5 Classification on the presence of Metabolites

This section is based on learning neural networks to distinguish the cyst spectra on the presence of a single metabolite. In our attempt we focused on glucose, a metabolite under observation when analyzing a cyst spectrum, and glutamine. The reason for choosing these two metabolites are that they show an easy detectable class of metabolite levels, e.g. the ones with and the ones without the presence of that metabolite.

Using this approach we increased our dataset from 9 training spectra and 3 test spectra to 23 training spectra and 1 test spectrum, again using the so called *leave one out* method. The reason for taking only one test spectrum was because we wanted to maintain an as large as possible training set.

5.5.1 Choosing the Network

Since we're still dealing with the classification paradigm, again the feed forward neural networks are used in combination with the back-propagation learning paradigm. The network contained three layers, one input, one hidden, and one output layer.

Input Layer

The input layer contains 47 input units, each representing a certain metabolite level as described in [1]. The range of the units was again from 0.0 to 500.0.

Hidden Layer

The number of units taken for the hidden layer was determined by experiment. The number of units was taken variable and the performance of the network was tested by the squared sum error. Figure 5.5 shows the squared sum error for the various number of hidden units.

Because of the fact that learning took quite a long time, about 2 days per training session, in this experiment, the number of hidden units was taken fixed after studying the error-curves.

It showed that the network was capable of solving the problem when about 40 hidden units were used. This value was taken for further investigation.

Output Layer

The final purpose of the network was to determine whether or not a certain metabolite was present in the input spectrum. Since this only requires two values, 0 = not present, 1 = present, we only need one output unit. The output value of a specific spectrum immediately gives us a measure of probability on the appearance of the metabolite in question.

5.5.2 Activation Rule

We used the same activation rule as in the previous experiment, e.g. the log-sigmoid rule. This activation rule appeared to learn much faster then the tan-sigmoid rule.

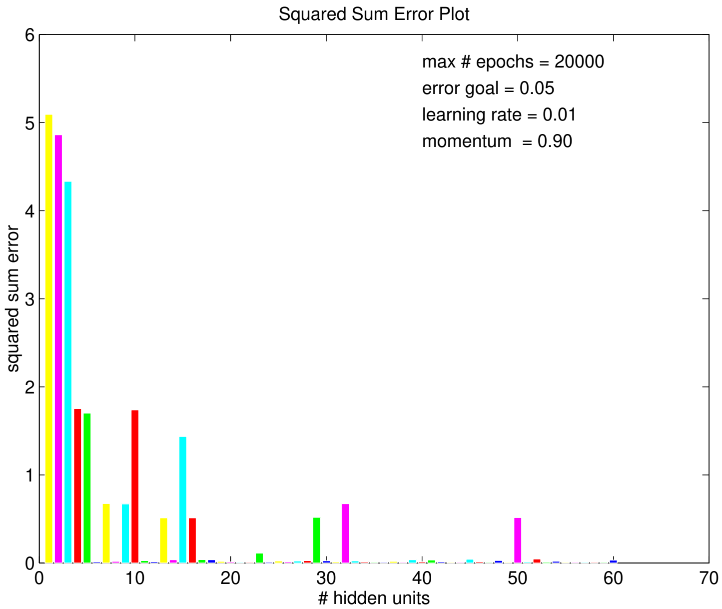


Figure 5.5: The SSE plotted against the number of hidden units.

5.5.3 Parameters

The parameters used were slightly different from the ones we used in the previous experiment. The next subsections will describe each of them.

Learning Rate and Momentum

Finding proper values for both learning rate and momentum is just a matter of experimenting. As mentioned in the chapter about neural networks good values for both parameters are those at which the network learns the fastest en doesn't get stuck in local minima.

After some training sessions we found the following values, learning rate = 0.01 and momentum 0.90.

It must be said that even with these values there is no guarantee that the network will always reach its error goal.

The Squared Sum Error Goal

The squared sum error goal was also determined by experiment, and turned out to be near 0.05.

Initial Weights and Biases

The initial weights and biases were taken in the range of [+1,-1], and the network performed reasonably well, e.g. no extra scaling of the weights was necessary.

5.5.4 Results

As mentioned before, we did two experiments, one testing for the presence of glutamine and one for glucose. The metabolite was said to be present in the spectrum if its metabolite level was greater than zero.

For both glutamine and glucose, this came down to a distribution of 16 spectra showing the presence of the metabolite in the spectrum, and 8 spectra with a metabolite level of zero. A significant difference between the two metabolites was that the range of the glutamine levels was [0.0,4.3] and for glucose the values were in the range of [0.0,17.9]. All spectra indicating the presence of glucose had a value greater than 1, while the spectra indicating the presence of glutamine had 6 spectra with glutamine levels in the range of [0.4,0.9].

In both experiments the parameter values were the same. The network was trained 35 times with different training- and test sets, which were chosen randomly.

Training Results

The training results for both glucose and glutamine were excellent, from the 35 training sessions there were no output values below 0.85, which means that each spectrum in the training set was learned correctly. Table 5.1 shows the training results in percentage.

Metabolite	$0.75 \leq n \leq 1.0$	$0.5 < n < 0.75$	$n \leq 0.50$
glucose	100%	0%	0%
glutamine	100%	0%	0%

Table 5.1: Training results for glucose and glutamine for 35 sessions, n represents the output value given by a specific spectrum.

The interval [0.75,1.0] is taken as a measure for absolutely correct classification, while spectra with output values in the range of (0.5,0.75] are probably correct, and values below 0.50 are definitely false classified.

Of 35 training sessions for glucose there were 22 sessions in which one of the spectra in the training set had a value between 0.85 and 0.90, the other 13 training sessions all had values in the range of [0.90,0.99]. For glutamine there were 11 sessions which had one spectrum in the range of [0.85,0.90) and 24 sessions in the range of [0.90,0.99].

Test Results

After training the network was presented a test spectrum, one which was not part of the training set. Table 5.2 shows the results for both glucose and glutamine.

Metabolite	$0.75 \leq n \leq 1.0$	$0.5 < n < 0.75$	$n \leq 0.50$
glucose	43%	17%	40%
glutamine	46%	6%	48%

Table 5.2: Test results for glucose and glutamine for 35 sessions, n represents the output value given by a specific spectrum

As can be seen from this table, the number of correct classifications hardly differs for both metabolites. In the class of probably correct classifications, we see that glucose has 17% probably correct, while glutamine only has 6% probably correct. The most common reason for this can be found in the metabolite levels of glucose, since in all spectra the presence of glucose had a metabolite level greater than 1, the distinction between presence- and non-presence of glucose could be made easier then in the case of glutamine, which had 6 spectra with a metabolite level in the range of [0.4,0.9].

5.5.5 Conclusions

Based on the results of this experiment, it can be said that, although the training set is still relatively small, the network doesn't generalize very well on these kinds of data.

There are a number of reasons why this can be the case:

- The training set isn't large enough.
- The network can't handle this kind of data, e.g. preprocessing steps must be taken to create a dataset, which can be learned ideally.This could for example be PCA, which reduces a spectrum to the components that characterize the spectrum, resulting in input patterns that can optimally be divided in input-space.

- The network learned too much specific features, this might explain the good training results. One way to overcome this problem is by reducing the number of hidden units, but in that case learning becomes more difficult.
- The network learned to little, resulting in poor generalization.

As came forward from the experiment with total spectra, lots of training data are needed to handle these kinds of problems, if they can be handled this way at all.

In order to say something valuable about neural networks being used as diagnostic methods, we need to know more about how and what the network is learning. Since the neural network appears to us as a black box, no inside information is presented. The way to gather this kind of knowledge is by experimenting with for example simulated data.

5.6 References

[1] P. van Vierzen, L. Massuger, U. Engelke, R. Wevers, J. Ruijs and A. Heerschap
High Resolution ^1H Magnetic Resonance Spectroscopy of Benign and Malignant Ovarian Cyst Fluids
ISMRM 1996, p1159

[2] Wevers R.A., Engelke U., Wendel U., de Jong J.N.G., Gabreels and Heerschap A.
A Standardized method for high-resolution ^1H -NMR of cerebrospinal fluid
Clin. Chem., 1995;41:744-751

[3] Warren S. Sarle
Stopped Training and Other Remedies for Overfitting
To appear in Proceedings of the 27th Symposium on the Interface, 1995
SAS Institute Inc.

[4] James L. McClelland, David E. Rumelhart
EXPLORATIONS IN PARALLEL DISTRIBUTED PROCESSING, A Handbook of Models, Programs, and Exercises
The MIT Press 1988

[5] Lawrence S. Anker and Peter C. Jurs
Prediction of Carbon-13 Nuclear Magnetic Resonance Chemical Shifts

by Artificial Neural Networks
Anal. Chem. **1992**, 64, 1157-1164

[6] James R. Long, Howard T. Mayfield, and Michael V. Henley
Pattern Recognition of Jet Fuel Chromatographic Data by Artificial Neural Networks with Back-Propagation of Error
Anal. Chem. 1991, 63, 1256-1261

[7] Yrjö Hiltunen, Esko Heiniemi, and Mika Ala-Korpela
Lipoprotein-Lipid Quantification by Neural-Network Analysis of ^1H NMR Data from Human Blood Plasma
Journal of Magnetic Resonance, **B 106**, 191-194 (1995)

[8] James R. Long, Vasilis G. Gregoriou, and Paul J. Gemperline
Spectroscopic Calibration and Quantification Using Artificial Neural Networks
Anal. Chem. **1990**, 62, 1791-1797

[9] Barry J. Wythoff, Steven P. Levine, and Sterling A. Tomellini
Spectral Peak Verification and Recognition Using a Multilayered Neural Network
Anal. Chem. **1990**, 62, 2702-2709

[10] Ray L. Somorjai, Alexander E. Nikulin, Nic Pizzi, et al.
Computerized Consensus Diagnosis: A Classification Strategy for Robust Analysis of MR Spectra.
I. Application to ^1H Spectra of Thyroid Neoplasms
Magnetic Resonance in Medicine 33:257-263 (1995)

[11] S. Dwarakanath, C.D. Ferris, J.W. Pierre, R.O. Asplund, D.L. Curtis
A Neural Network Approach to the Early Detection of Cancer
Biomedical sciences instrumentation 30:239-243 (1994)

[12] J.-P. Usenius, S.Tuohimetsä, P. Vainio, M. Ala-Korpela, Y. Hiltunen, et al.
Classification of Human Brain Tumors by Neural Network Analyses Using In Vivo ^1H MRS Metabolite Phenotypes
SMRM 1995:1710

[13] M. Ala-Korpela, Y. Hiltunen and J.D. Bell
Absolute Quantification of ^1H MRS Data from Human Plasma Using Neural Network Analysis
SMRM 1995:1988

[14] Suresh Kari, Nancy J. Olsen, Jane H. Park
Artificial Neural Network Analysis of P-31 MR Spectroscopy Data for

Evaluation of Muscle Diseases: Input Parameter Weighting
SMRM 1995:361

Chapter 6

Classification of Simulated Data

6.1 Introduction

Based on the results of the previous experiments it's questionable if neural networks can learn spectral data, without some sort of preprocessing. The aim of this experiment is:

> *to investigate, whether neural networks can learn spectral data, and to get more inside information about what they learn.*

In order to do this we created a relatively large dataset containing spectral data and noise data. In this chapter we will describe our experiences with this approach and try to find solutions for the problems we encountered.

6.2 Requirements on Simulated Spectra

To keep the simulation as real as possible, the simulated data has to fulfill certain requirements.

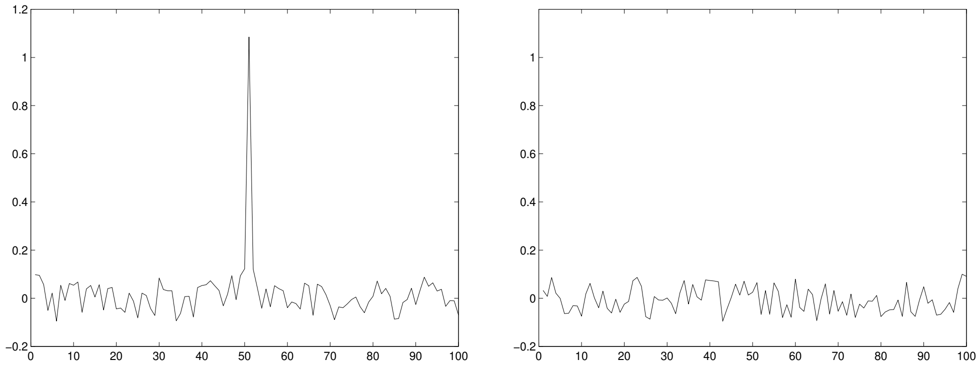
- the sample frequency must be at least $2 * f_{max}$ (Shannon's theorem)
- the SNR must be at least 1:3

6.2.1 The Dataset

We created a dataset based on the following function:

$$f(t) = asin(bt) + c.$$

The function f was then computed for 200 data points. Next the absolute value of the FFT of f was taken to form the spectral data. Since we're taking the FFT of a single sinus function, only one peak appears to be in the spectrum. The peak width and height are taken constant, the only distortion of the spectra was due to the random noise, which was was added to the spectrum leaving at least a SNR of 1:3. Since the resulting spectrum is mirrored around the Nyquist frequency, we only took the first part of the spectrum. This resulted in spectra as shown in figure 6.1(a). Next to the computed spectra, we constructed noise data by just taking random values, that didn't contradict with our required SNR. Figure 6.1(b) shows an example of such a noise signal.



(a) Example of a simulated spectrum (b) Example of a noise signal

Figure 6.1: Examples of simulated data

By visual inspection these kinds of signals are easy to classify by means of a threshold value. It is expected that it shouldn't be much harder for neural networks either.

6.3 Training based on Simulated Spectra

What has to be trained in our case is the detection of the presence of a peak on a certain position. In order to do so we need a dataset containing noise data and spectral data. We created a training set of 100 spectra and 100 noise signals from a set of 400. The other 200, of which 100 are spectra and 100 are noise, are used as test data. The picking of the training set from the total dataset was done random.

If our network can be learned, some separation in feature space has to result as depicted in figure 6.2. Above a specific noise level a borderline between the area indicating the presence of the peak and the non-presence of the peak will result.

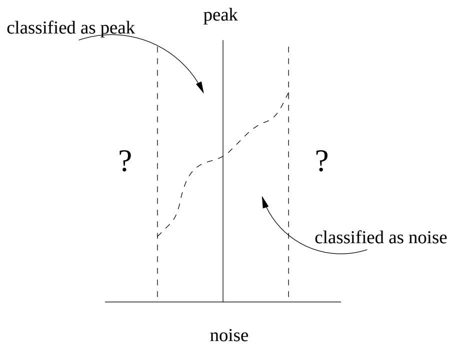


Figure 6.2: Learning area in a peak-noise situation, the question mark represents the area in which it's unknown how the network will respond to these kinds of data, when these conditions do not belong to the training of the data.

To measure the performance of the network, the test set must be in the same area as the training set, i.e. the same noise level, or else the results can not be predicted. This requirement was fulfilled since all test data came from the same set as the training data.

6.4 The Network

To gather more inside information about the back-propagation learning paradigm with respect to spectral data we used again a three layered feed forward neural network. The next subsections will describe each of the layers.

6.4.1 The Input Layer

The input layer consists of 100 input units, all with values in the range of [0:1.2]. First the simulated spectra were scaled between 0 and 1, then the random noise was added. This eventually came down to values in this particular interval.

6.4.2 The Hidden Layer

Like in all previous experiments, the number of hidden units to be used was a variable under observation. Since the detection of a peak in spectra like these is obviously a simple problem for a human observer, it is expected that not so many hidden units are required.

6.4.3 The Output Layer

The output layer contains only one unit, indicating the presence of the peak or not. The output value of this unit is also a measure of probability for the appearance of the peak, e.g. a measure of how well the peak could be detected.

6.4.4 The Activation Rule

As in the previous experiments, the activation rule is again the sigmoid rule. This rule seems to be the fastest for this kind of data.

6.4.5 Parameters

In contradiction with the previous experiments, it was very hard to find a set of parameters that would easily do the job. Neither one of the parameter settings used, succeeded in good performance. This means that for many training sessions, the parameter values were not adequate for reaching our goal.

We've tried several values for each of the parameters involved, but none gave very promising results. In the end we came up with the following values.

Learning Rate and Momentum

After quite some trials, although short in time, the value for the learning rate was taken 0.1 and the momentum 0.95. Varying the learning rate in steps of 0.05 didn't improve the performance on learning. Learning, if learned, went relatively fast, e.g. within 150 epochs. The network often got stuck at local minima.

The Squared Sum Error Goal

The SSE was again determined by the method described by Sarle [1], and was taken to be 0.01.

Initial Weights and Biases

The initial weights performed best when taken randomly between [-0.05,+0.05]. However, these values are far from an ideal learning situation, in which the network reaches its error-goal in almost all trials. This was with these initial weights not the case, nor with any other set of weights.

6.5 Results

Learning the training set gave quite some problems. The system didn't always reach the error goal and often got stuck at high local minima, e.g. minima with an SSE of about 110, or 100. In order to speed up the process of "getting results", we constantly watched the training process, by displaying the error values, and interrupted this process when a local minimum was entered, e.g. when the error stayed at a constant value which was not the required error-goal. The network was then started again, with new random initial weights from the same set of values as before. This problem didn't vanish when other initial parameter values were used, or when the number of hidden units was increased. This indicates that the error surface of spectral data is rather jagged and contains many local minima.

The number of hidden units was first varied from 10 to 110 units in steps of 5 units. There were no signs of improved learning, nor the speed of the training process nor the results after training increased.

Note well: in those cases where the error goal was reached the training set was learned for 100%, which means that it is possible to learn the network as long as the "optimal" initial parameters are used.

The results for the training set were all in the range [0.996, 1.0] wich means a 100% correct result.

After training the network was presented the test data. The result of these tests are given in figure 6.3 As can be seen from figure 6.3, it didn't

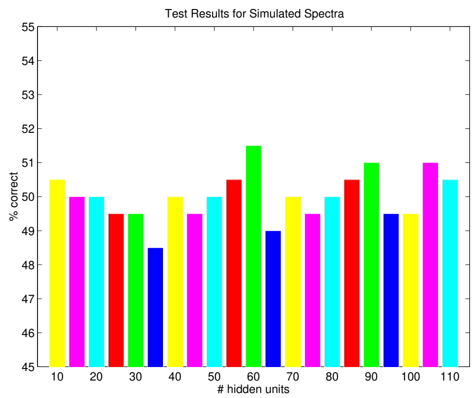


Figure 6.3: Test results for the simulated spectra for various number of hidden units.

quite matter how many hidden units we used, the percentage of correctly classified spectra was dangling about 50%. Even when less than 10 hidden units were used there was no change in the performance on the test set, the training set however still learned very well.

6.6 Conclusion

Although in literature most experiments using neural networks involve some sort of preprocessing of the input data, mainly because of the lack of computer facilities, this experiment shows that preprocessing of the input data may be essential when dealing with the kind of signals, like NMR-spectra.

It's hard to say why the network can't learn this kind of data. One of the most common explanations is that the peak is filtered out, e.g. could not be captured by the weights, which results in a sort of peak filtering network instead of a peak detection network.

Another reason could be that we still haven't presented enough learning data, in which case the generalization of the network couldn't be fulfilled.

6.7 References

[1] Warren S. Sarle
Stopped Training and Other Remedies for Overfitting
To appear in Proceedings of the 27th Symposium on the Interface, 1995
SAS Institute Inc.

[2] David E. Rumelhart, James L. McClelland and the PDP research group
Parallel Distributed Processing
Explorations in the Microstructure of Cognition
Volume 1: Foundations
1986 , The Massachusetts Institute of Technology

[3] David E. Rumelhart, James L. McClelland and the PDP research group
Parallel Distributed Processing
Explorations in the Microstructure of Cognition
Volume 2: Psychological and Biological Models
1986 , The Massachusetts Institute of Technology

[4] James L. McClelland, David E. Rumelhart
EXPLORATIONS IN PARALLEL DISTRIBUTED PROCESSING,
A Handbook of Models, Programs, and Exercises
The MIT Press 1988

Chapter 7

Conclusions

Although the results of the experiments do not look very promising. It's hard to say in this stage that neural networks can't be used for spectrum analysis at all. Previous research in this field already showed that neural networks in combination with PCA can lead to promising results, e.g. a correctly classified percentage of over 80%.

What these experiments do show is that:

- the ff-network with the back-propagation learning rule is not capable of learning these kinds of "spectral" data
- some sort of preprocessing on the spectral data has to be done in order to make a good division of the input patterns in input-space

Besides the general issues, referring to the case of spectral data, something can be said about the application of neural networks. During the experiments some very important factors appeared that need to be considered when people want to use these networks for special purposes:

- in the first case you need lots of training data, with a small training set it is hard to make any judgments about the results obtained
- training neural networks with many input/hidden units requires a lot of computational power, e.g. it takes a lot of time (days instead of minutes)
- the data has to be consistent, as in the ^{1}H cyst fluids a standardized method was used to obtain the data

To use neural network for the analysis of spectra more research is required. This research could be focussed on different kinds of neural networks and/or the application of preprocessing tools in combination with neural networks.

Glossary

apodization techniques	Techniques to improve the appearance of the spectrum.
artificial neural network	A collection of nodes that pass information to one another to compute a certain function. Artificial neural networks try to simulate the functioning of the biological brain.
baseline	The horizontal axis at which the spectrum is located.
carrier frequency	The frequency of RF pulse used.
chemical shift	Variation in the nominal Larmor frequency for a particular isotope within the molecular site. The amount of chemical shift is proportional to the magnetic field strenght and is usually specified in parts per million (ppm) of the actual resonant frequency relative to a standard resonance frequency.
gyro magnetic constant	Constant of proportionality relating to the angular frequency of precession of a nucleus to the magnetic field strenght: the value is both constant and specific to a particular isotope and is expressed in Mhz/Tesla.
in vitro	*in vitro* experiments refer to experiments on tissues outside the living system.
in vivo	*in vivo* experiments refer to experiments under which the living system is functioning, e.g. the "normal" situation.
line broadening	Spectral lines become broader, e.g. a specific nucleus contributes at a number of very closely spaced frequencies, according to chemical shift anf field homogeinity.

longitudinal relaxation Return of the component of the macroscopic magnetization vector along the static magnetic field, after excitation, to its equilibrium value by exchange of energy between the nuclear spins and the lattice.

magnetic moment A measure of the net magnetic properties of an object or particle. A nucleus with an intrinsic spin will have an associated magnetic dipole moment, so that it will interact with a magnetic field (as if it were a tiny bar magnet)

MRI Magnetic Resonance Imaging

MRS Magnetic Resonance Spectroscopy

NMR Nuclear Magnetic Resonance

nuclear coupling Magnetic interaction among proton groups.

quadrature phase detection A detection method at which the FID is sampled in the xy-plane in both x and y direction, resulting in a phase shift of 90^o degrees.

SI Spectroscopic Imaging, a technique which lays a raster over a certain area of which the spectra have to be taken, it then determines the spectrum of each field in the raster which gives a spectroscopic image of the area under investigation.

spectral width The frequency range, from f_{min} to f_{max}.

spin-lattice relaxation See longitudinal relaxation.

spin-spin relaxation See transverse magnetization.

transverse relaxation Component of the macroscopic magentization vector at right angles to the static magnetic field.

zero filling A technique applied to improve the visual appearance of the spectrum, zeros are added to the end of the FID which is then Fourier Transformed. The result is an interpolated spectrum.

Index

— **W** —

— **Z** —